EECS427 Final Report

Group 3: In-SRAM Boolean Operations

Group members: Yu-Sheng Ting (yushengt), Cheng-Lin Lee (iamjohn), Xinxin Wang (xinxinw),
I-Kang Wu (ikangwu), Yufan Gao (yfgao), Ganrun Xu (ganrunxu)

Abstract

To reduce the data movement over the memory hierarchy, we present an SRAM design that enables in-SRAM computation. This SRAM is at the same memory level as DMEM and supports four additional instructions following specific timing constraints. All transistors are carefully sized for READ/WRITE operations. It can avoid suffering from data disturbing during NOR/AND operations with special duty cycle adjustment. The SRAM functional test results are available.

Introduction

Speed and efficiency are the two most strong impetuses of computing development. In the baseline processor, a large fraction of time and energy is spent moving data over hierarchy. We present a SRAM design based on the conventional processor for reducing data movement through in-SRAM processing using bit line computing. Our key idea is to transform SRAM cells into computational units and enable computation without transferring data in or out of it. This optimization is at the expense of design complexity and extra area cost. The SRAM needs to cooperate with a modified sense amplifier and a SRAM controller.

Design and Implementation

1. System Overview of In-memory Computation

   Fig.1 shows the block diagram of the CPU controller and SRAM. After an SRAM-accessing instruction is decoded, the CPU controller will signal the SRAM controller. The SRAM controller will then take corresponding actions.

   In this work, we have two memory modules in the same hierarchical level. The DMEM module is used

for regular load/store operations, while the SRAM module is customized to support in-memory computation. The SRAM module can support regular load/store operations as well, but such operations correspond to special operation codes to distinguish themselves from load/store to DMEM module (see Additional ISA support for details).

2. Additional ISA Support

In this work, four additional instructions are added, and the specifications are shown in Table1. SRAM_AND and SRAM_NOR are the two in-memory computation instructions, and the operands are specified using SRAM address. SRAM_READ loads the value from SRAMaddr to Rdest; SRAM_WRITE stores the value from Rsrc into SRAMaddr.

Mnemonic	15~12 (OpCode)	11~8	7~4 (OpCodeExt)	3~0
SRAM_AND	0000	SRAMaddr	0100	SRAMaddr
SRAM_NOR	0000	SRAMaddr	1000	SRAMaddr
SRAM_READ	0000	Rdest	1100	SRAMaddr
SRAM_WRITE	0000	Rsrc	1111	SRAMaddr

Table 1. Additional ISA support

3. Timing Specification

Since each in-memory computation takes 2 cycles (details in the next section), an in-memory computation instruction must be followed by a regular instruction. If an SRAM-related instruction goes after an in-memory computation instruction, a NOP must be inserted between them.

4. In-memory Logic Computation

Fig.2(a) shows the basic bit line computation technology (AND and NOR) for SRAM. For AND operation, bitline (BL) is firstly precharged and then two worldline rows are activated. If both two bits are high, the bitline and the sensed result stays at high. If one or both of these activated bits are low, stored value on BL will gradually make a high to low transition, while the sensed results will change to low quickly once the voltage on BL drops below a certain threshold voltage (Vref). Similarly, a NOR operation could be

performed by sensing the result of bitline-bar(BLB).

5. Detail Implementation and Transistor Sizing

We have designed a 16x16 SRAM array. To guarantee the read/write operation correctness, we must carefully size the transistors in the SRAM cell and the precharge/discharge circuit.

6T SRAM cell (schematic shown in Fig.2(b)) is chosen to minimize the area of the design. To maintain the read stability, the width of N1/N3 should be larger than N2/N4. To reduce the leakage current, we need to size the length of N2/N4 slightly larger than the other transistors. To maintain the write stability, the width of the PMOS should be smaller than N2/N4.

In the first half of each cycle, the precharge transistors would turn on to precharge the BL to VDD. WE should be high in precharge phase for write operation, which would result in the contention of P2 and N1, as shown in Fig.2(c). Thus, to make sure the BL/BLB can be discharged to low voltage when Data/Data_b is high, the discharge NMOS should be wider than the precharge PMOS.

In our design, we choose an analog based sense amplifier connected with a buffer. Size of P1 is equal to the size of P2, which works as a current mirror. And size of N1 should be small since it functions as a current source, and smaller N1 means lower current and lower power consumption. After precharged, the output will stay at high or make a high to low transition. Therefore, N2 should be stronger than P2.

As shown in Fig.3, when two WLs are activated in AND/OR operations, the data stored in the two activated cells could be disturbed. This problem can be solved by shortening the duty cycle of the WL activation signal. The data ramp up or down would be small enough to recover by the cross-coupled inverters. Meanwhile, the BL voltage drop can be large enough to drive the sense amplifier to 0 output.

Result and Conclusion

1. SNM Analysis

To study the static noise margin of the designed SRAM cell, we plot the butterfly curve as shown in Fig.4. In the analysis, we swept the temperature from -55℃ to 125℃. We observe that the noise margin would decrease as the temperature varies. A more thorough analysis for the design robustness should include the consideration of process variation.

2. NC-Verilog Simulation Result

The integrated SRAM is tested in the following two schemes, read&write and NOR&AND. For the first scheme (see Fig.5(a), it generally writes and reads consecutively with data 0 and 1. For the second scheme (see Fig.5(b), we store two values in different words, execute an in-memory computation operation and finally read the computed value.

3. Layout Result

The layout of the whole design is shown in Fig.6. We use a vertical power stripe in the middle of the design, and block power rings for IMEM and DMEM. The core size is $441.98\mu m \times 549.8\mu m$ ($242901.64\mu m^2$), and the chip size (with IO pads) is $1264\mu m \times 1170\mu m$ ($1478880\mu m^2$).

4. Conclusion

In the baseline processor, some applications may need keeping moving data over the memory hierarchy. Our custom SRAM can help eliminate parts of data movements by supporting in-SRAM boolean operations, and hence reduce the overall execution time and energy consumption. Furthermore, we size the SRAM cells carefully and introduce a specialized clock signal with the duty cycle of 25%, so that the operations work correctly without causing data disturbing issues. Finally, we propose a specification for the SRAM to be integrated with the baseline processor, and simulate the design to verify the functionality.

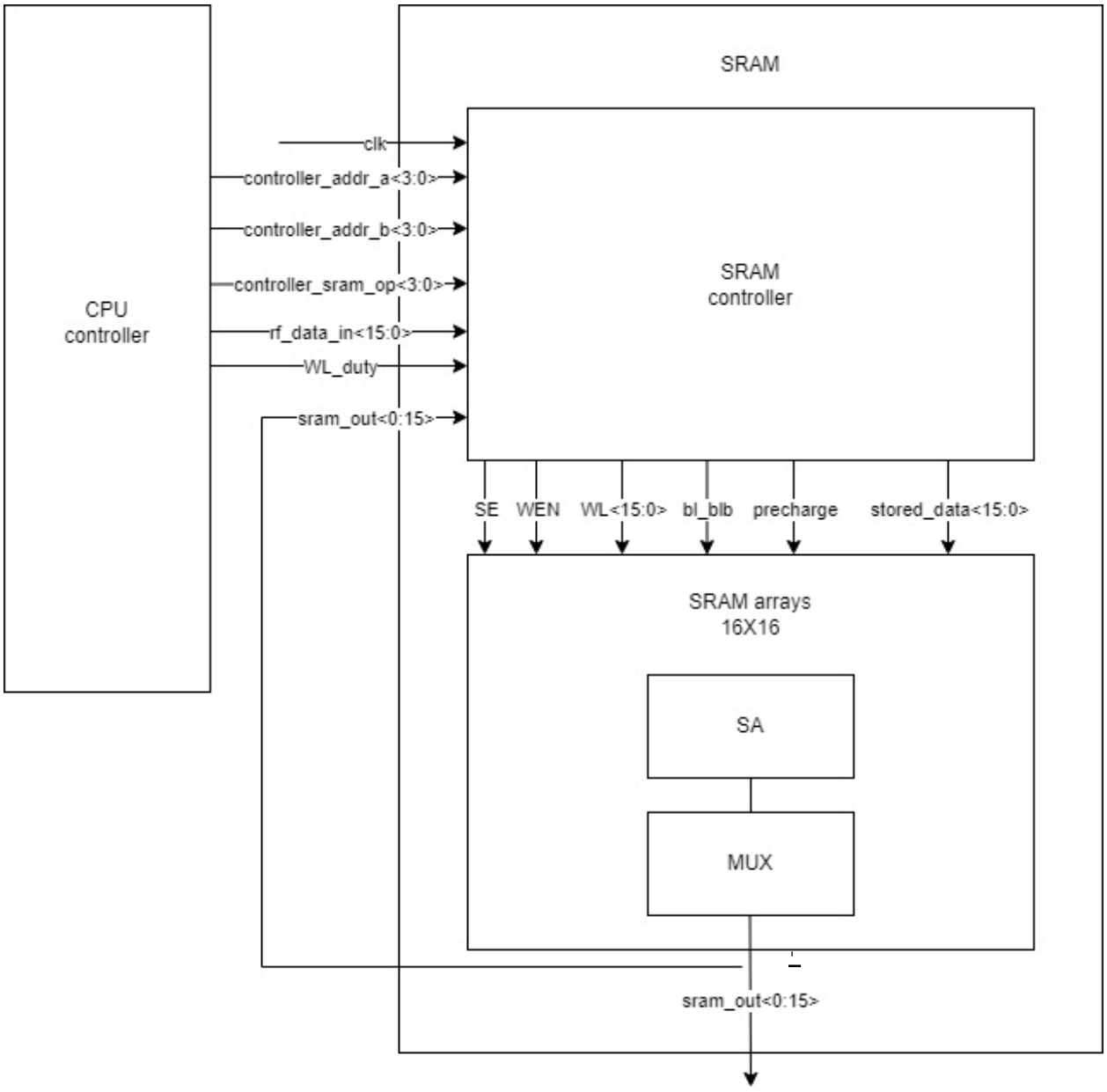


Fig. 1 SRAM block diagram

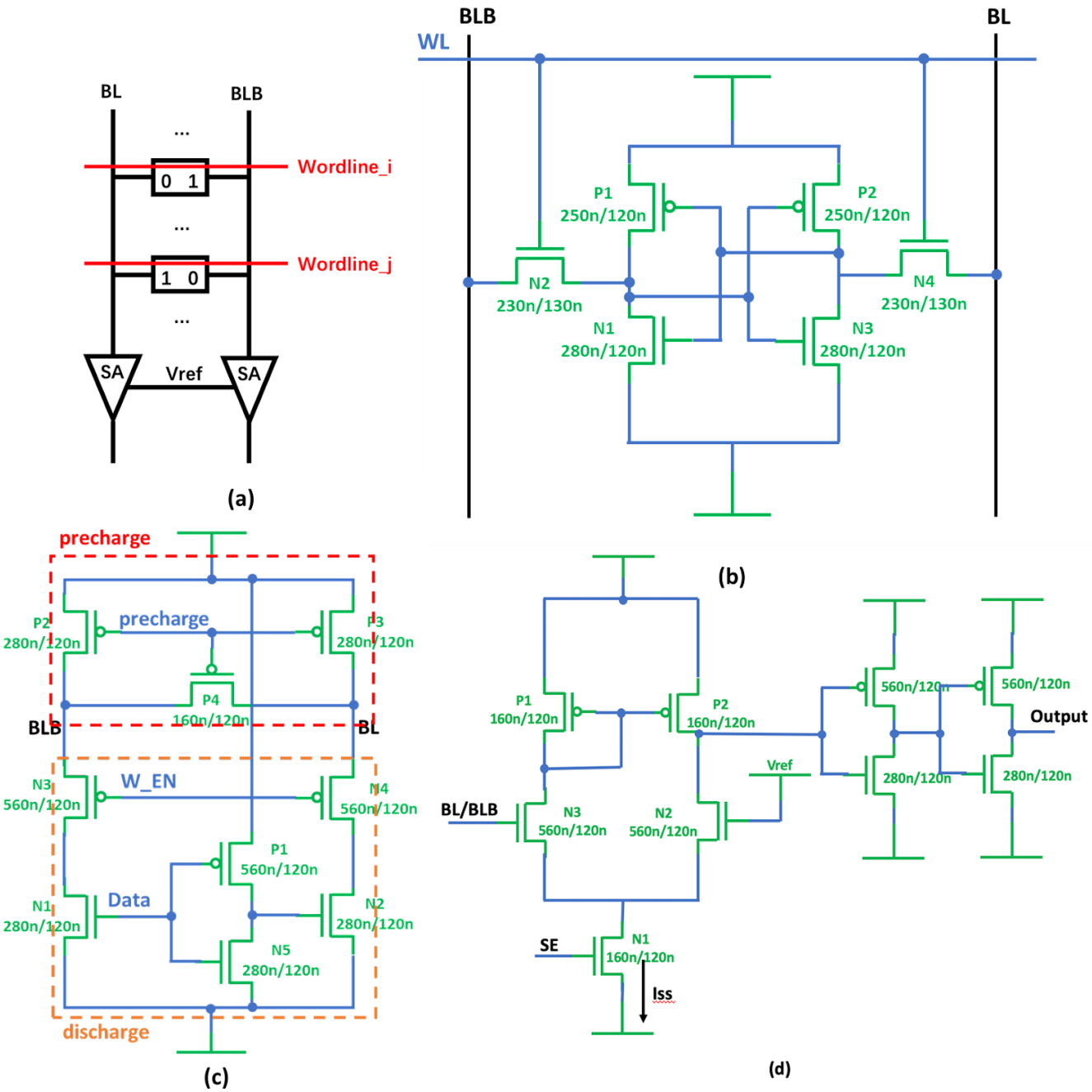


Fig. 2: (a) SRAM circuit for AND and NOR operation. (b) Schematic of SRAM cell, (c) precharge and discharge circuit and (d) sense amplifier.

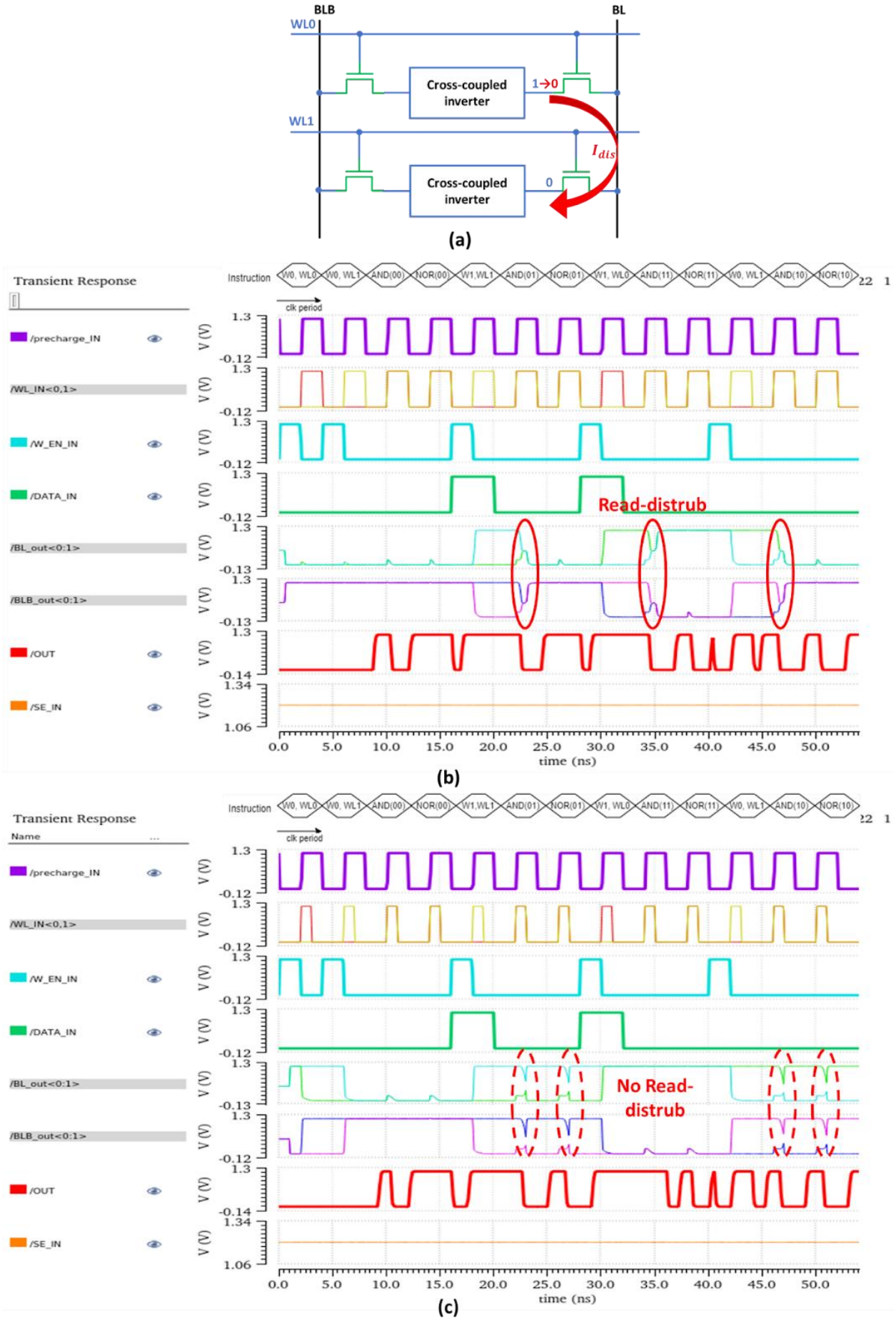


Fig. 3 (a) The mechanism of read-distrub. (b) The waveforms showing read-distrub. (c) By narrowing the pulse width of WL activation down to half of the precharge signal, the read-distrub can be avoided.

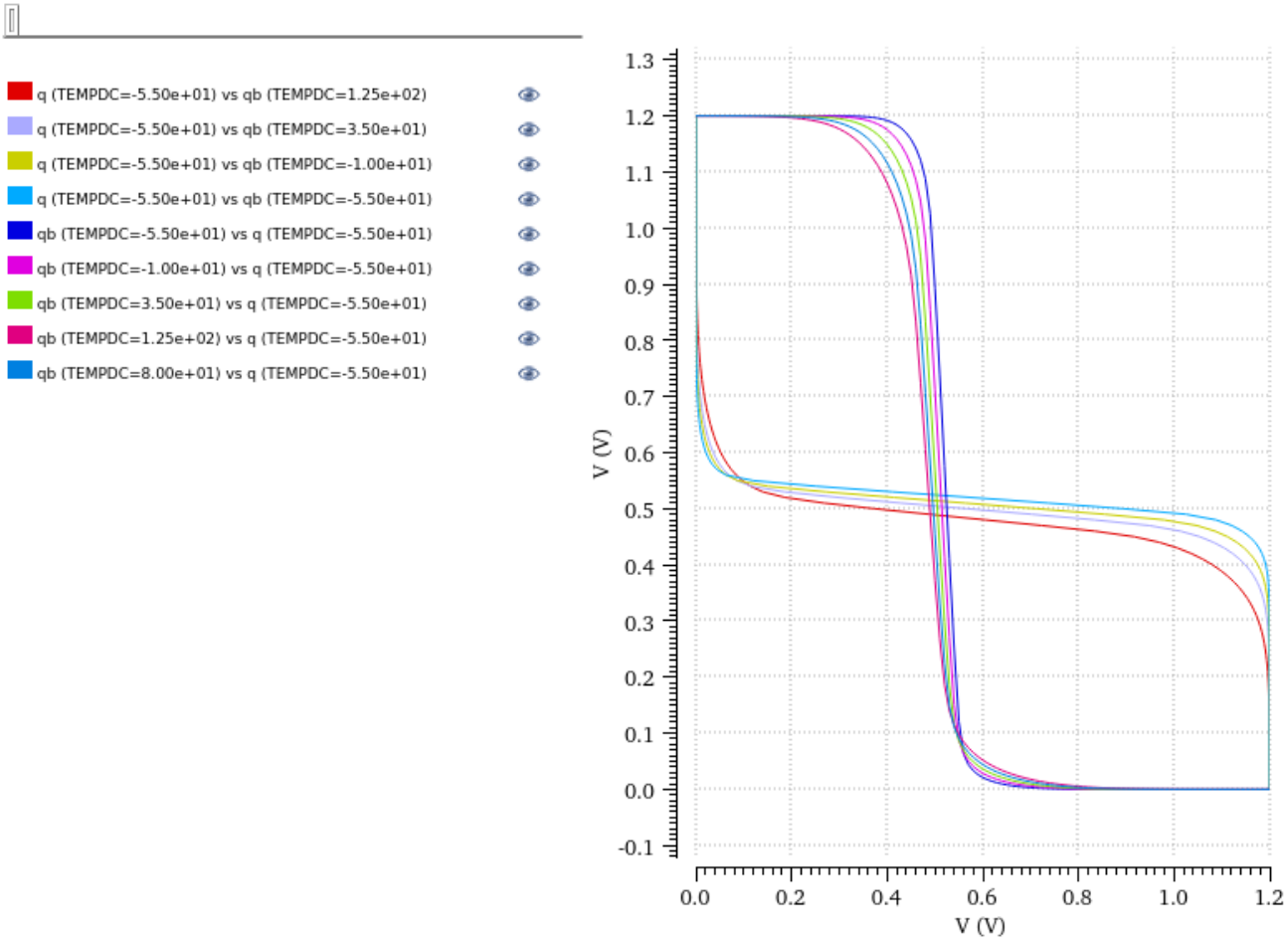
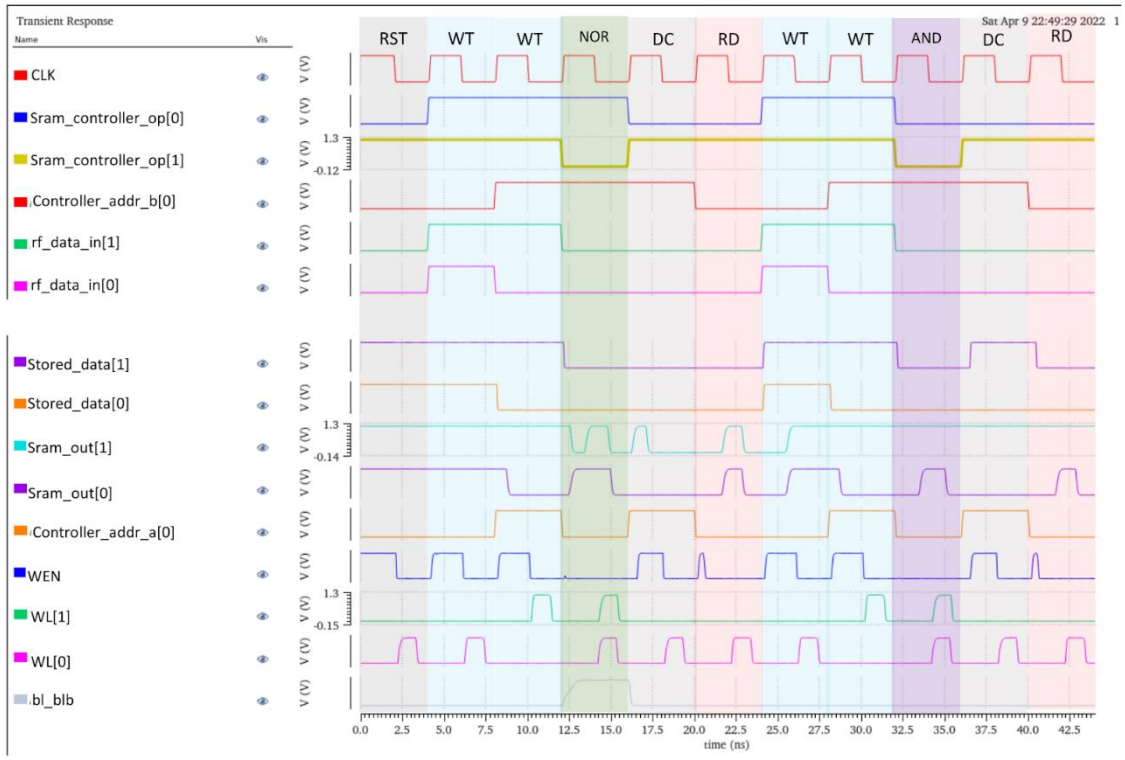


Fig. 4 Static noise margin analysis.

(a)



(b)

Fig 5. Spice simulation result for the scheme (a) read & write (b) NOR & AND.

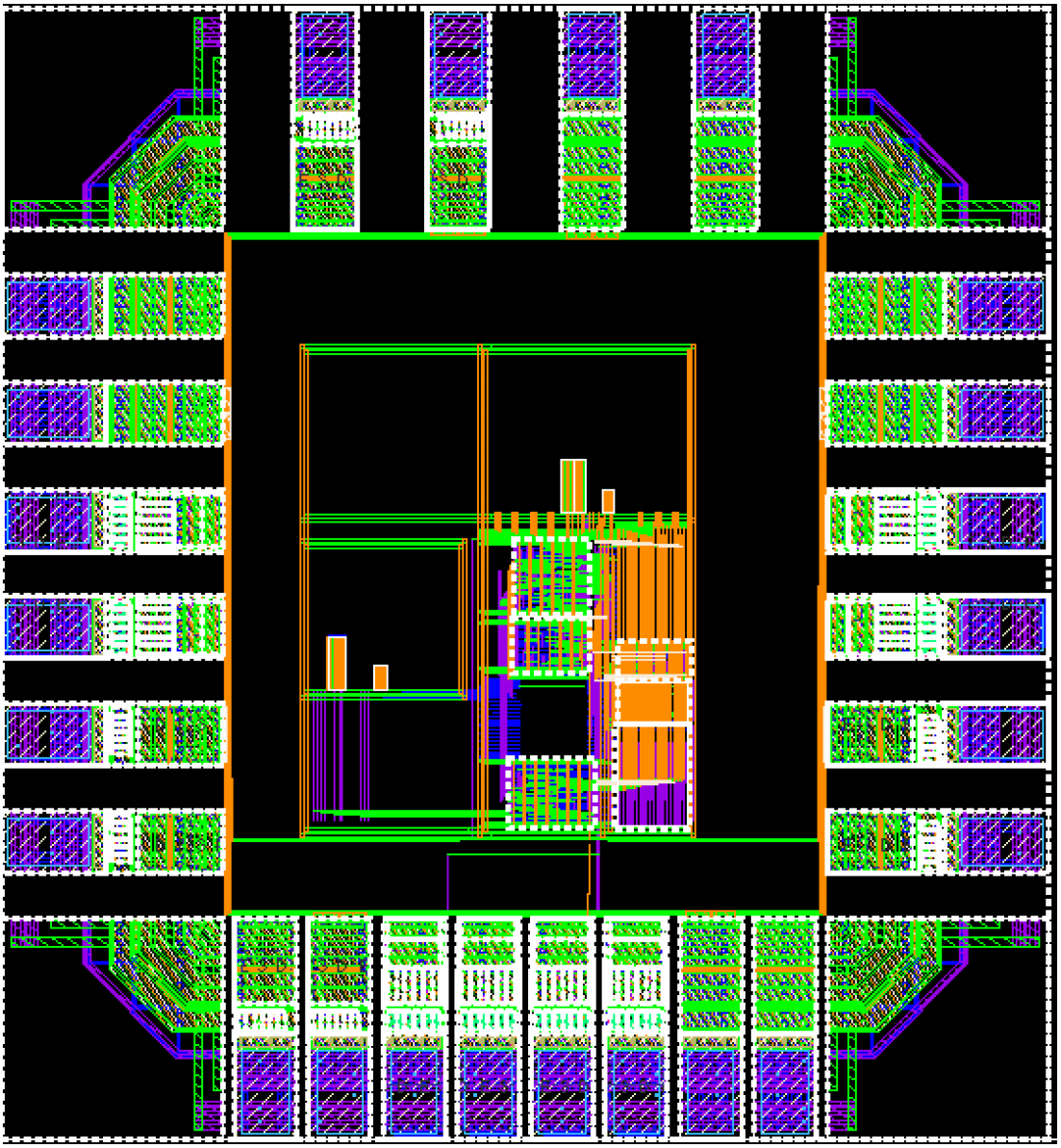


Fig 6. Layout of the integrated design.

Reference

[1] S. Aga, S. Jeloka, A. Subramaniyan, S. Narayanasamy, D. Blaauw and R. Das, "Compute Caches," 2017 IEEE International Symposium on High Performance Computer Architecture (HPCA), 2017, pp. 481-492, doi: 10.1109/HPCA.2017.21.